

09/820,433

MS158551.1

AMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

- 1) (Previously Presented) A system for bridging disparate object systems, comprising:
 - a memory boundary between a managed and an unmanaged object system;
 - a first interface wrapper to bridge communications across the memory boundary between a first object system and a second object system; and
 - a second interface wrapper to bridge communications across the memory boundary between the second object system and the first object system, wherein the first interface wrapper insulates the first object system from interface implementations in the second object system and the second interface wrapper insulates the second object system from interface implementations in the first object system to facilitate transparent communications between the first and second object systems.
- 2) (Original) The system of claim 1, wherein the first object system is at least one of a managed object system and an unmanaged object system, and the second object system is at least one of a managed object system and an unmanaged object system.
- 3) (Original) The system of claim 1, wherein the first wrapper and second wrapper serve as a proxy to the respective object systems that point to a stub within the wrappers in order to marshal data between the object systems.
- 4) (Original) The system of claim 1, wherein the first wrapper queries for type information from the second object system and forms interfaces from methods exposed from the type information.

09/820,433

MS158551.1

- 5) (Original) The system of claim 1, wherein the second wrapper calls the first object system and determines an interface by casting to a type and examining an exception.
- 6) (Original) The system of claim 5, wherein an adapter object is provided to map interfaces of an unknown type in the first object system to a known type in the second object system.
- 7) (Original) The system of claim 1, wherein the first object system is reference counted and the second object system is traced.
- 8) (Original) The system of claim 7, wherein the first wrapper maintains a traced reference for the second object system and reference counts interfaces utilized by the first object system.
- 9) (Original) The system of claim 7, wherein the second wrapper holds a traced reference for the second object system and releases interfaces utilized by the first object system.
- 10) (Original) The system of claim 7, further comprising a garbage collector to reclaim objects within the second object system, wherein unmanaged objects are reclaimed based upon the reference count in the first object system.
- 11) (Original) The system of claim 1, wherein object identities are maintained by utilizing a single managed wrapper per each object.
- 12) (Original) The system of claim 11, wherein a specialized wrapper is defined that subtypes off of a generic wrapper to simulate a class.

09/820,433

MS158551.1

-
- 13) (Original) The system of claim 1, further comprising a bridging services component to detect an unmanaged interface call and direct a managed client to an unmanaged object.
 - 14) (Original) The system of claim 13, wherein the unmanaged interface call is detected through a vtable reference from the second object system.
 - 15) (Original) The system of claim 1, wherein one or more objects belonging to the first and second object systems are activated *via* at least one of an early bound and late bound manner.
 - 16) (Original) The system of claim 15, wherein a late bound interface is employed to provide late bound activation.
 - 17) (Original) The system of claim 15, wherein early binding is provided at compile time *via* type information derived from a foreign object system.
 - 18) (Original) The system of claim 17, wherein type information is provided from at least one of a type library export and type library import tool.
 - 19) (Original) The system of claim 1, wherein the first object system utilizes results returned on a method call and the second object system utilizes exceptions.
 - 20) (Original) The system of claim 19, wherein results are mapped to exceptions and exceptions are mapped to results.
 - 21) (Original) The system of claim 1, wherein object reusability is provided *via* an inner object and outer object relationship.

09/820,433

MS158551.1

- 22) (Original) The system of claim 1, wherein intra object communications is provided *via* wrappers.
- 23) (Original) The system of claim 22, wherein inter object communications is provided *via* proxies within the wrappers.
- 24) (Original) The system of claim 1, wherein calls are routed to a foreign object system according to environment partitioning rules of the foreign object system.
- 25) (Original) A computer-readable medium having computer-executable components for executing the system of claim 1.
- 26) (Previously Presented) A method for bridging objects, comprising:
activating an interface wrapper from a first object system according to interface implementations of a second object system; and
utilizing the interface wrapper to facilitate transparent communications between managed and unmanaged object systems.
- 27) (Original) The method of claim 26, further comprising,
providing bridging services to direct the communications between the object systems.
- 28) (Original) The method of claim 26, further comprising,
proxying the respective object systems from a stub within the wrappers in order to marshal data between the object systems.
- 29) (Original) The method of claim 26, further comprising,
querying type information from the second object system; and
forming interfaces from methods exposed from the type information.

09/820,433

MS158551.1

- 30) (Original) The method of claim 26, further comprising,
determining an interface by casting to a type; and
examining an exception resulting from the caste.
- 31) (Original) The method of claim 26, further comprising,
maintaining object identities by utilizing a single managed wrapper per
each object.
- 32) (Original) The method of claim 31, further comprising,
creating a specialized wrapper that subtypes off of a generic wrapper to
simulate a class.
- 33) (Original) The method of claim 26, further comprising,
activating objects *via* at least one of an early binding and a late binding.
- 34) (Original) The method of claim 26, further comprising,
providing type information from at least one of a type library export and
type library import tool.
- 35) (Original) The method of claim 26, further comprising,
mapping results to exceptions; and
mapping exceptions to results.
- 36) (Original) The method of claim 26, further comprising,
routing calls to a foreign object system according to environment
partitioning rules of the foreign object system.

09/820,433

MS158551.1

- 37) (Previously Presented) A system for bridging objects, comprising:
means for activating an interface wrapper from a first object system
according to interface implementations of a second object system; and
means for utilizing the interface wrapper to facilitate transparent
communications between managed and unmanaged object systems.
- 38) (Original) The system of claim 37, further comprising,
means for directing communications between the object systems.
- 39) (Original) The system of claim 37, further comprising,
means for proxying the respective object systems in order to marshal data
between the object systems.
- 40) (Previously Presented) A signal facilitating object communications, comprising:
a signal for communicating between managed and unmanaged object
systems;
an interface wrapper activated *via* the signal from a first object system
according to interface implementations of a second object system, wherein the interface
wrapper facilitates transparent communications between the one or more object systems.
- 41) (Original) The signal of claim 40, wherein the signal is communicated over at
least one of a network system and a wireless system.

09/820,433

MS158551.1

- 42) (Previously Presented) An object system bridge, comprising:
at least one interface wrapper; and
a bridge service to enable the interface wrapper and facilitate transparent communications between at least one of a managed object system and an unmanaged object system;
wherein the interface wrapper insulates the at least one of the managed object system and the unmanaged object system from interface implementations in at least one other managed object system and unmanaged object system.